Assignment #3 (*Computer Security and Privacy*) [Mahavir Jhawar]
Submission Deadline: April 16, 2018
Marks: 40

Write a **python** program to implement the following variant of the ElGamal public key encryption scheme.

- **KeyGen**: This algorithm proceed as follows,
  - Choose two primes p and q such that $q = 2p+1$, and p is a 300-bit prime. [**See Note 1**]
  - Choose a $g \in \mathbb{Z}_q^*$ such that $< g >= \mathbb{QR}_q$.
  - Choose a number a uniformly at random from the set $\{2, 3, o(g) - 1\}$.
  - Compute $h = g^a \bmod q$. [**See Note 2**]
  - Set, $\mathsf{PK} = (q, g, h)$; $\mathsf{SK} = a$.
- **Encrypt**: To encrypt a message $m \in \mathbb{QR}_q$ under the public key $\mathsf{PK} = (q, g, h)$, this algorithm proceeds as follows:
  - Choose a random element $r \in \{2, \ldots, q-1\}$
  - Compute $C_1 = g^r \bmod q$
  - Compute $C_2 = (h^r \times m) \bmod q$
  - Output, ciphertext $C = (C_1, C_2)$
- **Decrypt**: To decryt a ciphertext $C = (C_1, C_2)$ under the secret $\mathsf{SK} = a$, this algorithm proceeds as follows:
  - Compute $t_1 = (C_1)^a \bmod q$.
  - Compute $t_2 = (C_2 \times t_1^{-1}) \bmod q$ [**See Note 3**]
  - Output decrypted message as t_2.

Some important instructions:

1. **Note1:** For prime generation, you must use Miller-Rabin Algorithm as described in Algorithm 5.7 of the reference book. [School book algorithm will take million of years to generate a 300-bit prime!!]
2. **Note2:** Use square-and-multiply algorithm given in Algorithm 5.5 of the reference book.
3. **Note3:** For modular inverse computation, use extended Euclidean algorithm given in Algorithm 5.2 of the reference book.
4. You must write and submit three independent programs - **KeyGen**, **Encrypt**, and **Decrypt**.
5. You might like to write this program in a way such that it is easily adaptable to the change of underlying group. After the submission of this program, you might be asked to write another variant where the underlying group is a multiplication subgroup of a Galois Field.

6. This program is also important in the sense that you will later be given to implement state-of-the-art digital signature scheme where you can call many built-in functions from this program.
7. Proficiency of the program include representation of the public key using standard encoding scheme such as ASN.1 (See `https://en.wikipedia.org/wiki/Abstract_Syntax_Notation_One`)
8. **Efficiency check will include measuring KeyGen, Encrypt and Decryption time.**
9. **35 marks, out of 40, are for correctness and efficiency. The rest 5 will account for programming proficiency.**